Distributed algorithms for fault tolerance

Consensus and related problems

Simin Nadjm-Tehrani

From last lecture

- Atomic broadcast can not be proved correct in an asynchronous network even in presence of a single crash failure!
- Why? What is the significance?

The consensus problem

- Processes $p_1,\dots,p_n$ take part in a decision
- Each p_i *proposes* a value v_i
- All correct processes *decide* on a common value v that is equal to one of the proposed values

Desired properties

- Every correct process eventually decides a value (Termination)
- No two correct processes decide differently (Agreement)
- If a process decides **v** then the value **v** was proposed by some process (Validity)

Common denominator

- Appears in many distributed problems, e.g. the Non-Blocking Atomic Commit (NBAC)
- At the end of a computation processes have to **commit** local computations (if things went well) and **abort** (if things went wrong)

NBAC problem

- At the end of a local computation the process votes YES or NO
- Then there is a decision procedure to commit or abort all local computations
- Commit: if all processes are correct and voted YES
- Abort: otherwise

NBAC properties

- Termination and Agreement as before
- Corresponding to validity:
  - If a process decides **Commit** then all processes have voted YES
  - If all processes have voted YES then the decision is **Commit**

Reduction to Consensus
[Guerraoui et al 2000]

Every process p_i executes

```
compute_locally()
" p_j send(vote) to p_j
await (delivery of a NO vote
       or ($ p_j suspected to be not correct)
       or delivery of a YES vote from each p_j)
If a NO vote delivered
   or a p_j suspected not correct
then v_i := abort else v_i := commit
propose(v_i), decide()  /* execute consensus */
```


Basic impossibility result

[Fischer, Lynch and Paterson 1985]
There is no deterministic algorithm solving the consensus problem in an asynchronous distributed system with a single crash failure

Partial correctness

Consider a bi-value decision:

- No configuration reachable from initial configuration has more than one decision value
- For each value ***v*** (0 or 1) there is an accessible configuration that has value ***v***

Total Correctness

- Partial correctness

Together with:

- Every admissible[1] run of the protocol is a deciding run

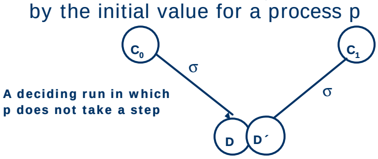
1)admissible run is a run in which only a single process is allowed to take finite number of steps, and messages are not lost

Proof sketch

- Assume total correctness
- Derive contradiction: Show that there is always a possibility for the protocol to remain indecisive
  - First, show there is some initial configuration from which deciding configurations with both values are possible (bi-valent)
  - Second, show there is a run in which no commitments are ever made

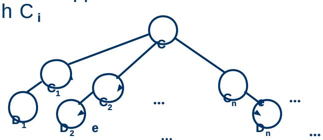
Part 1: initially bi-valent

- Assume not true
- By partial correctness both a 1-valent and 0-valent initial configuration must exist. Consider two such configurations differing only by the initial value for a process p



Part 2: indecisiveness

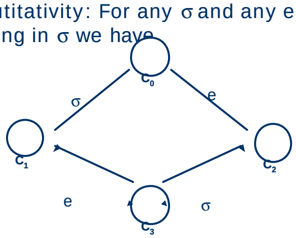
- Let C be a bi-valent configuration and $e=p^m$ an event applicable in C, then there is a bi-valent D_i reachable from some C_i provided that e was not applied to reach C_i



- Use commutativity of events!

Events commute

- Commutitativity: For any σ and any e not appearing in σ we have



Part 2: Core of proof

Assume all configurations D_i are not bi-valent, show contradiction!

- First, show that there exist both 0-valent and 1-valent configurations among D_i
- Finally show that there exists a bi-valent configuration A reachable from C_i

D_i includes 0-valent and 1-valent configurations

- Let E_i be an i-valent configuration reachable from C.

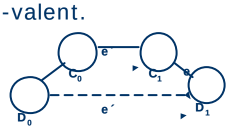
  Let $X = \{c \mid c = C_i\}$ and $D = \{c \mid c = D_i\}$.
- Either $E_i \in X$ or not.
  - If $E_i \in X$ then $e(E_i) \in D$, and is not bi-valent by assumption, so $e(E_i)$ is i-valent.
  - If $E_i \notin X$ then e was applied in reaching E_i, i.e. there is a configuration D_i from which E_i is reachable. Since elements of D are not bi-valent by assumption, then D_i is i-valent.

A bi-valent configuration A is reachable from some C_i

- Let C_i be neighbours differing by one event, and $e(C_i)$ be i-valent.

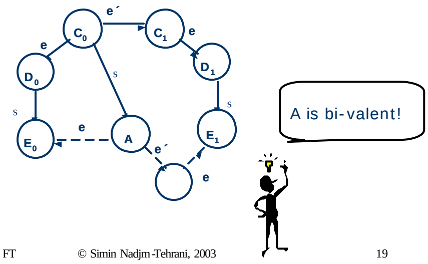
Two cases:

- e and e´ belong to different processes



- e and e´ belong to the same process

Case 2: Let σ be a sequence of deciding steps in which p is silent.



Reading material

- Chapter 14 in Tel's book (specifically section 14.1)
- Article by Fischer et al., Journal of ACM, 1985
- Article by Guerraoui, Hurfin et al. LNCS 1752, 2000